Collaborator: yc2454 (Yalu Cai)

1. Let us negate the goal and assume the negated goal is true to prove the contrapositive, which is logically equivalent to the original statement. The negated goal would be “ an NP-complete problem that has a algorithm.” We assume this to prove that SAT will have a algorithm. Since any NP-complete problem can be reduced to any other NP-complete problem, SAT can be reduced to in polynomial time, and so we could solve SAT through such a reduction. The total time associated with such a reduction plus the time to solve is , where means polynomial time in , which we can express as or , where and are natural numbers. We now need to show is . is anyway, so we can ignore it, and is still , so we only need to show is . The two sides have the same base –– 2, so we only **need to show is .**

If a function is , that by definition means , which means . Hence, . To show that is , we need to show . We already showed that as approaches infinity, the numerator approaches ; and it’s obvious that the denominator approaches infinity, so the whole thing approaches . is thus , and so is . So has a algorithm. The contrapositive is thus proven. QED.

1. We can solve by solving the part of it through brute force. For sake of simplicity, let , then . Brute forcing is of runtime . To show that has a algorithm, we need to show is , which through definition of little o notation means we need to show , such that , . Substituting in we get that **we need to show, such that ,** . Now take logs of both sides:

Observe that because grows faster than , so such a that we mentioned. Hence, has a algorithm.

1. Let us assume the negated goal to prove the contrapositive, i.e. . This is equivalent to (SAT has aalgorithm, which means we have to prove both and . For the second one, we will prove the original direction

Claim 1: ( )

Proof: If , it means there’s a polynomial time algorithm to solve , i.e. an algorithm where is a natural number and is the length of the formula. We can solve problems using this algorithm by padding with the correct number of ’s, which is a polynomial time reduction. The corresponding formula for an formula of size is of size . Then we use the supposed algorithm, and the runtime on this will be for some natural number . We want to show this is . To show this, we need to show such that . Take the log of both sides:

Claim 2: ( )

Proof: Using what we proved in the previous two parts of the question, we know that if doesn’t have any algorithms, then no NP-complete problem has a algorithm (part 1), but *does* have a algorithm, which means it is not NP-complete. QED.